COURSE PROJECT PART 3 (Final)

In Part 3 you will implement a web application that supports the following features. **You must use the table definitions that I will post (derived from the ERD that I posted earlier) Unless you need to make some small additions/modifications to support you additional features.** If you do modify the table definitions, you will be responsible for translating the test data we will provide so that it matches your table definitions.

REQUIRED Application Use Cases(Functionalities):

l)View Public Info: All users, whether logged in or not, can

- See the meetups that are occurring during some date range.
- select an interest and see list of groups that share that interest

2)Login: User enters username, x and password y via forms on login page. This data is sent as POST parameters to the login-authentication component, which checks whether there is a tuple in the Person table with username=x and the password = md5(y) (You can use more secure Hash algorithms too.)

- If so, login is successful. A session is initiated with pid stored as a session variables. Optionally, you can store other session variables. Control is redirected to a component that displays the user's home page.
- If not, login is unsuccessful. A message is displayed indicating this to the user.

3)Display Home Page: Once a user has logged in, MeetUp will display her home page. Also, after other actions or sequences of related actions, are executed, control will return to component that displays the home page. The home page should display

- Error message if the previous action was not successful.
- Any other information you'd like to include. For example, you might want to include groups the user belongs to, events she's signed up for, etc. on the home page, or you may prefer to just show them when she does some of the following use cases.

After Logging in successfully a user may do any of the following use cases:

4)View My Upcoming Events: Provide various ways for the user to see upcoming events for which she has RSVP'd (saying she's attending). The default should be showing events for the current day and the next three days. Optionally you may include a way for the user to specify a range of dates. In each case,_ the display should include the e_id, date, start, Iname, zip, and name of sponsoring group.

S)RSVP for an event: User chooses an event and signs up for the event. You may find it easier to implement this along with a use case to search for events. If the user has

previously signed up for the event no additional action is needed (but MeetUp should not crash).

6) **Create an event:** If the user is authorized to do so, he or she creates a new event for a group, providing all the needed data, via forms. The application should prevent unauthorized users from doing this action.

7) ***Logout:*** *The session is destroyed and a "goodbye" page or the login page is displayed*

ADDITIONAL FEATURES

You will propose at additional use cases. (Suggestions ... various ways to search for events of interest, search for groups of interest, join group, rating events, creating members, creating groups, granting ability to create events, etc.)

Some of these should involve modifying inserting and/or updating rows in some table(s) and also involve helping users find events or information about them.

Additional Requirements:

- You will need to bring the host computer to the demo/test session at the end of the semester..
- **Enforcing complex constraints:** Your MeetUp implementation should prevent users from doing actions they are not allowed to do. For example, Meetup should prevent users who are not authorized by group creator from organizing events for the group. This should be done by querying the database to check that the user belongs to the group, and is authorized by creator before allowing him to create/alter the event. You may also use the interface to help the user to avoid violating the constraints. For example, you could provide a list of groups and events for which the user is eligible and allow to make changes. However, you should not rely solely on client-side interactions to enforce the constraint, since a user could bypass the client-side interface and send malicious http requests.
- When a user logs in, a session should be initiated; relevant session variables should be stored. When the member logs out, the session should be terminated.
- Each component executed after the login component should authenticate the session and retrieve the user's pid from a stored session variable.
- **You must use prepared statements if your programming language supports them.** If your programming language does not support prepared statements, Free form inputs (i.e., text entered through text boxes) that is incorporated into SQL statements should be validated or cleaned to prevent SQL injection
- You should take measures to prevent cross-.site scripting vulnerabilities, such as passing any text that comes from users through htmlspecialchars or some such function, before incorporating it into the html that MeetUp produces.
- The user interface should be readable, but it does not need to be fancy.

- For help in debugging and evaluating your project, you may want to echo each query that the application executes to a file or to the html that the Meetup generates.
- **Extra features:** There are lots of opportunities to add functionality that makes Meetup better. You may add more features for fun. However, your grade will be based almost entirely on our evaluation of the basic features (including those that you propose). This will be done largely by executing a series of tests. Please don't add any bells and whistles that interfere with testing the basic features.

What you should do

1. Study my solution to Part 2. Drop the tables you defined in Part 2 then install the ones given in my solution to Part 2.
2. Before you start coding, think about what each component (URL) will do. If there are common features among many of the components, think about how you will modularize your code.
3. For each component
   - Using some sample data, write the queries executed by the component, and test them.
   - Code the component.
   - Test the component with additional values, including values that are not valid inputs.
4. Suggestion: Implement and test the components one at a time. When a component is ready, add links to it to your home page (or enhance the home page with the interface of the new component.) You will get partial credit if some of your features work, even if others have not been implemented.

PROGRESS REPORT A progress report will be due in about 2 weeks. In this report, you must propose the additional features that you plan to add and show code for use-cases. For team projects, you must demonstrate that each team member has written some of the code. You will hand this in electronically and I may also schedule brief meetings to discuss. Details ***TBD. THIS IS A MANDATORY PART OF THE PROJECT AND WILL AFFECT YOUR GRADE.***

Complete Final Project Hand-in instructions: You will hand in:

- A zip containing your source code, .sql file, css files, image files (if any) etc to run your project properly.
- Instructions for using your MeetUp system: the URL and anything that isn't obvious about how to use your MeetUp. Also instructions regarding how to setup your application using the files uploaded by you
- A list of the files in your application and what's in each file. (E.g. "homepage.php: script to generate home page".)
- A separate file that lists all of the use cases and the queries executed by them (with brief explanation). This should be well organized and readable. It should be detailed enough to give readers a good idea of how your application works, without making them dig through all the code.
- Brief description of additional features you added.
- For team projects: A summary of who did what.
- For team projects you will also hand in (individually, via another link to be provided) a brief self-assessment and an assessment of your teammate's work.

Shortly before the project is due, I'll ask you to sign up for a time slot to demonstrate your project to me or the TA.. This demo will mainly consist of running application on a bunch of test cases. We may also ask you to explain some of your code. No formal presentation will be expected.

Consider this as the final instructions. I will post additional instructions if i feel its necessary.

The total project grade will be 25% of your course grade. Part 1 counts for about 15% of the project grade. Part 2 counts for about 10 to 15% of the project grade. Part 3 counts for about 70 to 75% of the project grade. There may also be a quiz or exam question(s) based on the project.